Exhibit K8

1 of 2

NIST Special Publication 800-175B

Guideline for Using Cryptographic Standards in the Federal Government:

Cryptographic Mechanisms

Elaine Barker

This publication is available free of charge from:
http://dx.doi.org/10.6028/NIST.SP.800-175B

C O M P U T E R S E C U R I T Y

National Institute of Standards and Technology
U.S. Department of Commerce

NIST Special Publication 800-175B

Guideline for Using Cryptographic Standards in the Federal Government:

Cryptographic Mechanisms

Elaine Barker
Computer Security Division
Information Technology Laboratory

This publication is available free of charge from:
http://dx.doi.org/10.6028/NIST.SP.800-175B

August 2016

U.S. Department of Commerce
Penny Pritzker, Secretary

National Institute of Standards and Technology
Willie May, Under Secretary of Commerce for Standards and Technology and Director

Authority

This publication has been developed by NIST in accordance with its statutory responsibilities under the Federal Information Security Modernization Act (FISMA) of 2014, 44 U.S.C. § 3551 *et seq.*, Public Law (P.L.) 113-283. NIST is responsible for developing information security standards and guidelines, including minimum requirements for federal information systems, but such standards and guidelines shall not apply to national security systems without the express approval of appropriate federal officials exercising policy authority over such systems. This guideline is consistent with the requirements of the Office of Management and Budget (OMB) Circular A-130.

Nothing in this publication should be taken to contradict the standards and guidelines made mandatory and binding on federal agencies by the Secretary of Commerce under statutory authority. Nor should these guidelines be interpreted as altering or superseding the existing authorities of the Secretary of Commerce, Director of the OMB, or any other federal official. This publication may be used by nongovernmental organizations on a voluntary basis and is not subject to copyright in the United States. Attribution would, however, be appreciated by NIST.

National Institute of Standards and Technology Special Publication 800-175B
Natl. Inst. Stand. Technol. Spec. Publ. 800-175B, 73 pages (August 2016)
CODEN: NSPUE2

This publication is available free of charge from:
http://dx.doi.org/10.6028/NIST.SP.800-175B

Comments on this publication may be submitted to:

National Institute of Standards and Technology
Attn: Computer Security Division, Information Technology Laboratory
100 Bureau Drive (Mail Stop 8930) Gaithersburg, MD 20899-8930
Email: SP800-175@nist.gov

Reports on Computer Systems Technology

The Information Technology Laboratory (ITL) at the National Institute of Standards and Technology (NIST) promotes the U.S. economy and public welfare by providing technical leadership for the Nation's measurement and standards infrastructure. ITL develops tests, test methods, reference data, proof of concept implementations, and technical analyses to advance the development and productive use of information technology. ITL's responsibilities include the development of management, administrative, technical, and physical standards and guidelines for the cost-effective security and privacy of other than national security-related information in federal information systems. The Special Publication 800-series reports on ITL's research, guidelines, and outreach efforts in information system security, and its collaborative activities with industry, government, and academic organizations.

Abstract

This document is intended to provide guidance to the Federal Government for using cryptography and NIST's cryptographic standards to protect sensitive, but unclassified digitized information during transmission and while in storage. The cryptographic methods and services to be used are discussed.

Keywords

asymmetric-key algorithm; authentication; confidentiality; cryptography; digital signatures; encryption; integrity; key agreement; key derivation; key management; key transport; key wrapping; message authentication codes; non-repudiation; Public Key Infrastructure (PKI); random bit generation; symmetric-key algorithm.

Acknowledgments

The author wishes to thank the authors of SP 800-21 from which this document was derived, Annabelle Lee and William C. Barker, along with those colleagues that reviewed drafts of this document and contributed to its development: Lily Chen, Shu-jen Chang, and Kerry McKay. The author also gratefully acknowledges and appreciates the many comments from the public and private sectors whose thoughtful and constructive comments improved the quality and usefulness of this publication.

Table of Contents

SECTION 1: INTRODUCTION

1.1 Overview and Purpose

In today's environment of increasingly open and interconnected systems and networks and the use of mobile devices, network and data security are essential for the optimum safe use of this information technology. Cryptographic techniques should be considered for the protection of data that is sensitive, has a high value, or is vulnerable to unauthorized disclosure or undetected modification during transmission or while in storage.

Cryptography is a branch of mathematics that is based on the transformation of data and can be used to provide several security services: confidentiality, data integrity authentication, and source authentication, and also to support non-repudiation.

- *Confidentiality* is the property whereby sensitive information is not disclosed to unauthorized entities. Confidentiality can be provided by a cryptographic process called *encryption*.
- *Data integrity* is a property whereby data has not been altered in an unauthorized manner since it was created, transmitted or stored. The process of determining the integrity of the data is called *data integrity authentication*.
- *Source authentication* is a process that provides assurance of the source of information to a receiving entity; source authentication can also be considered as identity authentication (i.e., providing assurance of an entity's identity). A special case of source authentication is called *non-repudiation*, whereby support for assurance of the source of the information is provided to a third party.

This document is one part in a series of documents intended to provide guidance to the Federal Government for using cryptography to protect its sensitive, but unclassified digitized information during transmission and while in storage; hereafter, the shortened term "sensitive" will be used to refer to this class of information. Other sectors are invited to use this guidance on a voluntary basis. The following are the initial publications in the Special Publication (SP) 800-175 subseries. Additional documents may be provided in the future.

- SP 800-175A provides guidance on the determination of requirements for using cryptography. It includes the laws and regulations for the protection of the Federal Government's sensitive information, guidance for the conduct of risk assessments to determine what needs to be protected and how best to protect that information, and a discussion of the required security-related documents (e.g., various policy and practice documents).
- SP 800-175B (this document) discusses the cryptographic methods and services available for the protection of the Federal Government's sensitive information and provides an overview of NIST's cryptographic standards.

1.2 Audience

This document is intended for federal employees and others who are responsible for providing and using cryptographic services to meet identified security requirements. This document might be used by:

- Program managers responsible for selecting and integrating cryptographic mechanisms into a system;
- A technical specialist requested to select one or more cryptographic methods/techniques to meet a specified requirement;
- A procurement specialist developing a solicitation for a system, network or service that will require cryptographic methods to perform security functionality; and
- Users of cryptographic services.

The goal is to provide these individuals with sufficient information to allow them to make informed decisions about the cryptographic methods that will meet their specific needs to protect the confidentiality and integrity of data that is transmitted and/or stored in a system or network, as well as to obtain assurance of its authenticity.

This document is not intended to provide information on the federal procurement process or to provide a technical discussion on the mathematics of cryptography and cryptographic algorithms.

1.3 Scope

This document limits its discussion of cryptographic methods to those that conform to Federal Information Processing Standards (FIPS) and NIST Special Publications (SPs), which are collectively discussed as NIST "standards" in this document. While the Federal Government is required to use these standards, when applicable, industry and national and international standards bodies have also adopted these cryptographic methods.

This document provides information on selecting and using cryptography in new or existing systems.

1.4 Background

The use of cryptography relies upon two basic components: an *algorithm* and a *key*. The algorithm is a mathematical function, and the key is a parameter used during the cryptographic process. The algorithm and key are used together to apply cryptographic protection to data (e.g., to encrypt the data or to generate a digital signature) and to remove or check the protection (e.g., to decrypt the encrypted data or to verify the digital signature). The security of the cryptographic protection relies on the secrecy of the key. Security should not rely on the secrecy of the algorithm, as the algorithm specification may be publicly available.

In order to use a cryptographic algorithm, cryptographic keys must be "in place," i.e., keys must be established for and/or between parties that intend to use cryptography. Keys

may be established either manually (e.g., via a trusted courier) or using an automated method. However, when an automated method is used, authentication is required for the participating entities that relies on an established trust infrastructure, such as a Public Key Infrastructure (PKI) or on a manually distributed authentication key.

In general, keys used for one purpose (e.g., the generation of digital signatures) must not be used for another purpose (e.g., for key establishment) because the use of the same key for two different cryptographic processes may weaken the security provided by one or both of the processes. See Section 5.2 in SP 800-57, Part 1[1] for further information.

1.5 Terms and Definitions

The following terms and definitions are used in this document. In general, the definitions are drawn from FIPS and NIST Special Publications.

Algorithm	A clearly specified mathematical process for computation; a set of rules that, if followed, will give a prescribed result.
Approved	FIPS-Approved and/or NIST-recommended. An algorithm or technique that is either 1) specified in a FIPS or NIST recommendation, or 2) specified elsewhere and adopted by reference in a FIPS or NIST Recommendation.
Asymmetric-key algorithm	See *public-key algorithm*.
Authentication	A process that provides assurance of the source and integrity of information that is communicated or stored.
Bit string	An ordered sequence of 0's and 1's.
Block cipher algorithm	A family of functions and their inverse functions that is parameterized by cryptographic keys; the functions map bit strings of a fixed length to bit strings of the same length.
Certificate (or public key certificate)	A set of data that uniquely identifies an entity, contains the entity's public key and possibly other information, and is digitally signed by a trusted party, thereby binding the public key to the entity identified in the certificate. Additional information in the certificate could specify how the key is used and the validity period of the certificate.
Certificate Revocation List (CRL)	A list of revoked but unexpired certificates issued by a Certification Authority.

[1] SP 800-57 Part 1, *Recommendation for Key Management: General Guideline*.

Certification Authority (CA)	The entity in a public key infrastructure (PKI) that is responsible for issuing certificates and exacting compliance to a PKI policy.
Ciphertext	Data in its encrypted form.
Compromise	The unauthorized disclosure, modification, substitution or use of sensitive data (e.g., keying material and other security-related information).
Confidentiality	The property that sensitive information is not disclosed to unauthorized entities.
Cross certify	The establishment of a trust relationship between two Certification Authorities (CAs) through the signing of each other's public key in a certificate referred to as a "cross-certificate."
Cryptographic algorithm	A well-defined computational procedure that takes variable inputs, including a cryptographic key (if applicable), and produces an output.
Cryptographic boundary	An explicitly defined continuous perimeter that establishes the physical bounds of a cryptographic module and contains all the hardware, software and/or firmware components of a cryptographic module.
Cryptographic checksum	A mathematical value created using a cryptographic algorithm that is assigned to data and later used to test the data to verify that the data has not changed.
Cryptographic hash function	A function that maps a bit string of arbitrary length to a fixed-length bit string. **Approved** hash functions satisfy the following properties: 1. (One-way) It is computationally infeasible to find any input that maps to any pre-specified output, and 2. (Collision resistant) It is computationally infeasible to find any two distinct inputs that map to the same output.
Cryptographic key	A parameter used in conjunction with a cryptographic algorithm that determines its operation in such a way that an entity with knowledge of the key can reproduce or reverse the operation, while an entity without knowledge of the key cannot. Examples include: 1. The transformation of plaintext data into ciphertext data, 2. The transformation of ciphertext data into plaintext data,

	3. The computation of a digital signature from data, 4. The verification of a digital signature, 5. The computation of an authentication code from data, 6. The verification of an authentication code from data and a received authentication code, and 7. The computation of a shared secret that is used to derive keying material.
Cryptographic module	The set of hardware, software and/or firmware that implements **approved** security functions (including cryptographic algorithms and key generation) and is contained within a cryptographic boundary.
Cryptographic primitive	A low-level cryptographic algorithm used as a basic building block for higher-level cryptographic algorithms.
Cryptography	The discipline that embodies the principles, means and methods for providing information security, including confidentiality, data integrity, and non-repudiation.
Cryptoperiod	The time span during which a specific key is authorized for use or in which the keys for a given system may remain in effect.
Data integrity	A property whereby data has not been altered in an unauthorized manner since it was created, transmitted or stored.
Decryption	The process of changing ciphertext into plaintext using a cryptographic algorithm and key.
Digital signature	The result of a cryptographic transformation of data that, when properly implemented, provides the services of: 1. Source authentication, 2. Data integrity, and 3. Supports signer non-repudiation.
Digital Signature Algorithm (DSA)	A public-key algorithm that is used for the generation and verification of digital signatures.
Elliptic Curve Digital Signature Algorithm (ECDSA)	A digital signature algorithm that is an analog of DSA using elliptic curves.
Encryption	The process of changing plaintext into ciphertext for the purpose of security or privacy.

Entity	An individual (person), organization, device or process.
Ephemeral key pair	A short-term key pair that is generated when needed; the public key of an ephemeral key pair is not provided in a public key certificate, unlike static public keys which often are.
Function	As used in this document, used interchangeability with algorithm.
Hash function	See *cryptographic hash function*.
Hash value	The result of applying a hash function to information; also called a message digest.
Initialization Vector (IV)	A vector used in defining the starting point of a cryptographic process.
Integrity	The property that data has not been modified or deleted in an unauthorized and undetected manner.
Interoperability	The ability of one entity to communicate with another entity.
Key	See *cryptographic key*.
Key agreement	A (pair-wise) key-establishment procedure where the resultant secret keying material is a function of information contributed by two participants, so that no party can predetermine the value of the secret keying material independently from the contributions of the other party. Contrast with key-transport.
Key derivation	The process by which one or more keys are derived from either a pre-shared key, or a shared secret and other information.
Key establishment	The procedure that results in keying material that is shared among different parties.
Key management	The activities involving the handling of cryptographic keys and other related security parameters (e.g., IVs and counters) during the entire life cycle of the keys, including the generation, storage, establishment, entry and output, and destruction.
Key pair	A public key and its corresponding private key; a key pair is used with a public key (asymmetric-key) algorithm.
Key transport	A key-establishment procedure whereby one party (the sender) selects a value for the secret keying material and then securely distributes that value to another party (the receiver). Contrast with key agreement.

Key-wrapping key	A symmetric key used to provide confidentiality and integrity protection for other keys.
Keying material	The data (e.g., keys and IVs) necessary to establish and maintain cryptographic keying relationships.
Keying relationship, cryptographic	The state existing between two entities such that they share at least one cryptographic key.
Message Authentication Code (MAC)	A cryptographic checksum on data that uses a symmetric key to detect both accidental and intentional modifications of data.
Message digest	See *hash value*.
Mode of operation	An algorithm that uses a block cipher algorithm to provide a cryptographic service, such as confidentiality or authentication.
NIST standard	Federal Information Processing Standard (FIPS) or Special Publication (SP).
Non-repudiation	A service using a digital signature that is used to support a determination of whether a message was actually signed by a given entity.
Plaintext	Data that has not been encrypted.
Primitive	See *Cryptographic primitive*.
Private key	A cryptographic key, used with a public key cryptographic algorithm that is uniquely associated with an entity and is not made public. In an asymmetric (public) key cryptosystem, the private key is associated with a public key. Depending on the algorithm, the private key may be used to: 1. Compute the corresponding public key, 2. Compute a digital signature that may be verified by the corresponding public key, 3. Decrypt data that was encrypted by the corresponding public key, or 4. Compute a shared secret during a key-agreement process.
Public key	A cryptographic key used with a public-key cryptographic algorithm, that is uniquely associated with an entity and that may be made public. In an asymmetric (public) key cryptosystem, the public key is associated with a private key. The public key may be known by anyone and, depending on the

	algorithm, may be used to: 1. Verify a digital signature that is signed by the corresponding private key, 2. Encrypt data that can be decrypted by the corresponding private key, 3. Compute a shared secret during a key-agreement process.
Public key (asymmetric) cryptographic algorithm	A cryptographic algorithm that uses two related keys, a public key and a private key. The two keys have the property that determining the private key from the public key is computationally infeasible.
Public Key Infrastructure (PKI)	A framework that is established to issue, maintain and revoke public key certificates.
Relying party	An entity that relies on the certificate and the CA that issued the certificate to verify the identity of the certificate owner, and the validity of the public key, associated algorithms and any relevant parameters in the certificate, as well as the owner's possession of the corresponding private key.
RSA	A public-key algorithm that is used for key establishment and the generation and verification of digital signatures.
Secret key	A cryptographic key that is used with a symmetric (secret key) cryptographic algorithm and is not made public. The use of the term "secret" in this context does not imply a classification level, but rather implies the need to protect the key from disclosure. Compare with a private key, which is used with a public key algorithm.
Secret key (symmetric) cryptographic algorithm	See *symmetric (secret key) algorithm*.
Sensitive (information)	Sensitive, but unclassified information.
Security strength	A number associated with the amount of work (that is, the number of operations) that is required to break a cryptographic algorithm or system.
Shared secret	A secret value that is computed during a key-agreement transaction and is used as input to derive a key using a key-derivation method.

Signature generation	The use of a digital signature algorithm and a private key to generate a digital signature on data.
Signature verification	The use of a digital signature and a public key to verify a digital signature on data.
Source authentication	A process that provides assurance of the source of information.
Static key pair	A long-term key pair for which the public key is often provided in a public-key certificate.
Symmetric key	A single cryptographic key that is used with a symmetric (secret key) algorithm. Also called a secret key.
Symmetric (secret key) algorithm	A cryptographic algorithm that uses the same secret key for an operation and its complement (e.g., encryption and decryption).

1.6 Acronyms

AES	Advanced Encryption Standard; specified in FIPS 197.
ANS	American National Standard.
ANSI	American National Standard Institute.
ASC	Accredited Standards Committee.
CA	Certification Authority.
CBC	Cipher Block Chaining mode; specified in SP 800-38A.
CFB	Cipher Feedback mode; specified in SP 800-38A.
CKMS	Cryptographic Key Management System.
CP	Certificate Policy.
CPS	Certification Practice Statement.
CRL	Certificate Revocation List.
CTR	Counter mode; specified in SP 800-38A.
DES	Data Encryption Standard; originally specified in FIPS 46; now provided in SP 800-67.
DH	Diffie-Hellman algorithm.
DNSSEC	Domain Name System Security Extensions.
DRBG	Deterministic Random Bit Generator; specified in SP 800-90A.
DSA	Digital Signature Algorithm; specified in FIPS 186.
ECB	Electronic Codebook mode; specified in SP 800-38A.
ECDSA	Elliptic Curve Digital Signature Algorithm.
EMC	Electromagnetic Compatibility.
FCKMS	Federal Cryptographic Key Management System.
FIPS	Federal Information Processing Standard.
FISMA	Federal Information Security Management Act.
GCM	Galois Counter Mode; specified in SP 800-38D.
HMAC	Keyed-Hash Message Authentication Code; specified in FIPS 198.
IEC	International Electrotechnical Commission.

IEEE	Institute of Electrical and Electronics Engineers.
IETF	Internet Engineering Task Force.
EMI	Electromagnetic Interference.
INCITS	International Committee for Information Technology Standards.
IPSEC	Internet Protocol Security.
ISO	International Standards Organization.
IT	Information Technology.
MAC	Message Authentication Code.
MQV	Menezes-Qu-Vanstone algorithm; specified in SP 800-56A.
NRBG	Non-deterministic Random Bit Generator.
NIST	National Institute of Standards and Technology.
OFB	Output Feedback mode; specified in SP 800-38A.
OTAR	Over-the-Air-Rekeying.
PKI	Public Key Infrastructure.
RA	Registration Authority.
RBG	Random Bit Generator.
RFC	Request for Comment.
RSA	A public key algorithm attributed to Rivest, Shamir and Adleman.
SHA	Secure Hash Algorithm.
SP	Special Publication.
SSH	Secure Shell protocol.
TCG	Trusted Computing Group.
TDEA	Triple Data Encryption Algorithm; specified in SP 800-67.
TLS	Transport Layer Security.

1.7 Content

This document is organized into the following sections:

- Section 1 provides an introduction to the SP 800-175 series of publications and to this document in particular, and provides a glossary of terms and a list of acronyms.
- Section 2 discusses the importance of standards, as well as the national and international standards bodies concerned with cryptography.
- Section 3 introduces the **approved** algorithms used for encryption, digital signature and key-establishment, and provides discussions on security strengths and algorithm lifetime.
- Section 4 discusses the services that cryptography can provide: data confidentiality, data integrity authentication, source authentication and support for non-repudiation.
- Section 5 discusses the key management required for the use of cryptography, providing general guidance and discussions on key-management systems, key-establishment mechanisms and random bit generation.
- Section 6 discusses additional issues associated with the use of cryptography.

There is one appendix in this document:

- Appendix A lists applicable Federal Information Processing Standards, recommendations, and guidelines.

SECTION 2: STANDARDS AND GUIDELINES

2.1 Benefits of Standards

Standards define common practices, methods, and measures/metrics. Standards provide solutions that have been evaluated by experts in relevant areas, reviewed by the public and subsequently accepted by a wide community of users. By using standards, organizations can reduce costs and protect their investments in technology.

Standards provide the following benefits:

- **Interoperability.** Products developed to a specific standard may be used to provide interoperability with other products that conform to the same standard. For example, by using the same cryptographic encryption algorithm, data that was encrypted using vendor A's product may be decrypted using vendor B's product. The use of a common standards-based cryptographic algorithm is necessary, but may not be sufficient to ensure product interoperability. Other common standards, such as communications protocol standards, may also be necessary.

  By ensuring interoperability among the products of different vendors, standards permit an organization to select from various available products to find the most cost-effective solution.

- **Security.** Standards may be used to establish a common level of security. For example, most agency managers are not cryptographic security experts, and, by using an **approved** cryptographic algorithm and key length, a manager knows that the algorithm has been found to be adequate for the protection of sensitive government data and has been subjected to a significant period of public analysis and comment.

- **Quality.** Standards may be used to assure the quality of a product. Standards may:

  - Specify how a feature is to be implemented,
  - Require self-tests to ensure that the product is still functioning correctly, and
  - Require specific documentation to assure proper implementation and product-change management.

  Many NIST standards have associated conformance tests and specify the conformance requirements. The conformance tests may be administered by NIST-accredited laboratories and provide validation that the NIST standard was correctly implemented.

- **Common Form of Reference.** A NIST standard may become a common form of reference to be used in testing/evaluating a vendor's product. For example, FIPS 140[2] contains security and integrity requirements for *any* cryptographic module implementing cryptographic operations.

[2] FIPS 140, *Security Requirements for Cryptographic Modules*.

- **Cost Savings.** Implementations that comply with commonly accepted specifications provided by standards can save money. Without standards, users may be required to become experts in every information technology (IT) product that is being considered for procurement. Also, without standards, products may not interoperate with different products purchased by other users. This could result in a significant waste of money or in the delay of implementing IT.

2.2 Federal Information Processing Standards and Special Publications

2.2.1 The Use of FIPS and SPs

The use of a Federal Information Processing Standard (FIPS) is *mandatory* for the Federal Government whenever the type of service specified in that standard is required by a federal agency for the protection of sensitive information. For example, FIPS 197[3] contains a specific set of technical security requirements for the AES algorithm. Whenever AES is used by an agency, its implementation and use must conform to FIPS 197. A FIPS is **approved** by the Secretary of Commerce.

A NIST Special Publication (SP) is similar to a FIPS, but is not mandatory unless a particular government agency (e.g., OMB) makes it so. An SP does not need the approval of the Secretary of Commerce.

Although the requirements for the use of a FIPS and an SP are different, both types of publications have been subjected to the same review process by the federal agencies and the public. The approval process for a FIPS is more formal than that of an SP, and subsequently takes longer for the initial approval and the approval of any subsequent revisions.

When a federal agency requires the use of cryptography (e.g., for encryption), an **approved** algorithm must be used; approval is indicated by inclusion in a FIPS or SP. For example, two **approved** algorithms for encryption are AES (as specified in FIPS 197) and TDEA (as specified in SP 800-67[4]). Whenever encryption is used by a federal agency for the protection of sensitive information, either AES or TDEA must be used. Whenever AES is to be used, it must be implemented as specified in FIPS 197; whenever TDEA is to be used, it must be implemented as specified in SP 800-67. In addition to using **approved** algorithms, federal agencies are required to use only implementations of these algorithms that have been validated and are included in validated cryptographic modules (see Section 5.4.5 for further discussion).

When developing a specification or the criteria for the selection of a cryptographic mechanism or service, cryptographic algorithms specified in FIPS and SPs must be used, when available. Some guidelines may be used to specify the functions that the algorithm will perform (e.g., FIPS 199[5] or SP 800-53[6]). Other NIST standards specify the

[3] FIPS 197, *Advanced Encryption Standard (AES)*.

[4] SP 800-67, *Recommendation for the Triple Data Encryption Algorithm (TDEA) Block Cipher*.

[5] FIPS 199, *Standards for Security Categorization of Federal Information and Information Systems*.

[6] SP 800-53, *Security and Privacy Controls for Federal Information Systems and Organizations*.

operation and use of specific types of algorithms (e.g., AES, DSA) and the level of independent testing required for classes of security environments (e.g., FIPS 140).

Appendix A contains a list of FIPS and SPs that apply to the implementation of cryptography in the Federal Government. Note that when a FIPS is revised, its number is commonly followed by a revision number that indicates the number of times that it has been revised (e.g., "FIPS 186-4" is used to indicate the fourth revision of FIPS 186); this practice is not used in the main body of this document; the reader must refer to the latest version of the FIPS or SP that has been officially **approved** (see http://csrc.nist.gov/publications/; note that this site also contains clearly marked draft publications) .

2.2.2 FIPS Waivers

In the past, a waiver was sometimes issued by an agency to indicate that the use of a FIPS was not required by that agency. However, the Federal Information Security Management Act (FISMA) of 2002 (P.L. 107-347) eliminated previously authorized provisions for waivers from FIPS (see SP 800-175A for a discussion).

2.3 Other Standards Organizations

NIST develops standards, recommendations, and guidelines that are used by vendors who are developing security products, components, and modules. These products may be acquired and used by federal government agencies. In addition, there are other groups that develop and promulgate standards. These organizations are briefly described below.

2.3.1 American National Standards Institute (ANSI) [7]

The American National Standards Institute (ANSI) is the administrator and coordinator of the United States' private-sector voluntary standardization system. ANSI does not develop American National Standards itself; rather, it facilitates the development of standards by establishing consensus among qualified groups.

Several ANSI committees have developed standards that use cryptography, but the primary committee that has developed standards for the cryptographic algorithms themselves is Accredited Standards Committee (ASC) X9, which is a financial-industry committee[8]. Many of the standards developed within ASC X9 have been adopted within NIST standards (e.g., the Elliptic Curve Digital Signature Algorithm specified in American National Standard X9.62[9] has been adopted in FIPS 186); likewise, ASC X9 has approved the use of NIST standards via a registry of approved standards from non-ASC X9 sources (e.g., AES, as specified in FIPS 197).

A number of ASC X9 standards have also been incorporated into the standards of other standards bodies, such as the International Standards Organization (ISO) (see Section

[7] Further information is available at the ANSI web site: www.ansi.org.

[8] Further information is available at the ANSI X9 web site: x9.org.

[9] ANS X9.62, *Public Key Cryptography for the Financial Services Industry: The Elliptic Curve Digital Signature Algorithm (ECDSA)*.

2.3.4) via a Technical Advisory Group (TAG) called the International Committee on Information Technology Standards (INCITS). INCITS has been responsible for assuring that U.S. standards (e.g., both those developed by NIST and those developed within ASC X9) are incorporated within ISO standards.

2.3.2 Institute of Electrical and Electronics Engineers (IEEE) Standards Association[10]

IEEE is an international, professional association that is dedicated to advancing technological innovation and excellence. The technical objectives of the IEEE focus on advancing the theory and practice of electrical, electronics and computer engineering, and computer science. IEEE develops and disseminates voluntary, consensus-based industry standards involving leading-edge electro-technology. IEEE supports international standardization and encourages the development of globally acceptable standards.

The Institute of Electrical and Electronics Engineers Standards Association (IEEE-SA) is an organization within IEEE that develops global standards. It has more than one thousand active standards, some of which are related to cryptography.

IEEE P1363[11] is the only IEEE standard that focuses on cryptography. It includes a series of standards on public-key cryptography. IEEE P1363 was developed at the same time as the ANSI public-key cryptographic standards, such as ANS X9.31[12], X9.42[13], X9.44[14], X9.62[15], and X9.63[16], which were developed in ASC X9 (see Section 2.4.1).

- The first part of the IEEE P1363 standard was published in 2000 and revised in 2004 as IEEE P1363a[17]. It includes the basic public-key cryptography schemes, such as RSA encryption, signatures, the Digital Signature Algorithm (DSA), and key establishment using Diffie-Hellman (DH) and Menezes-Qu-Vanstone (MQV) over finite fields and elliptic curves.
- IEEE P1363.1[18], which was published in 2008, specifies NTRU encryption and signature schemes.

---

[10] Further information is available at the IEEE-SA web site: standards.ieee.org.

[11] IEEE P1363: *Standard Specifications for Public-Key Cryptography.*

[12] ANS X9.31, *Digital Signatures Using Reversible Public Key Cryptography for the Financial Services Industry (rDSA)*, which has now been withdrawn.

[13] ANS X9.42, *Agreement of Symmetric Keys Using Discrete Logarithm Cryptography*, which has now been withdrawn.

[14] ANS X9.44, *Key Establishment Using Integer Factorization Cryptography.*

[15] ANS X9.62, *The Elliptic Curve Digital Signature Algorithm (ECDSA).*

[16] ANS X9.63, *Key Agreement and Key Transport Using Elliptic Curve Cryptography.*

[17] IEEE P1363a, *Standard Specifications for Public Key Cryptography - Amendment 1: Additional Techniques.*

[18] IEEE P1363.1, *Public-Key Cryptographic Techniques Based on Hard Problems over Lattices.*

- IEEE P1363.2[19] was also published in 2008. It specifies password-authenticated key agreement and password-authenticated key retrieval schemes.

The schemes specified in IEEE P1363.1 and P1363.2 are not included in the NIST standards.

Cryptographic schemes are used in IEEE standards for different applications. One of the more notable is the IEEE 802 LAN/MAN group of standards, which are widely used computer networking standards for both wired (Ethernet) and wireless (IEEE 802.11[20]) networks. Cryptographic algorithms are used to protect wireless communications. The CCM mode for authentication and confidentiality specified in SP 800-38C was adopted from IEEE 802.11. Other AES modes of operation (e.g., GCM, which is specified in SP 800-38D) are also used in IEEE 802 standards. IEEE 802 standards also use the SHA-1 and SHA-2 family of hash functions specified in FIPS 180 and used in HMAC, as specified in FIPS 198.

XTS, a block cipher mode of operation specified in SP 800-38E, was adopted from IEEE P1619[21] as SP 800-38E.

2.3.3 Internet Engineering Task Force (IETF)

The Internet Engineering Task Force (IETF) is an international community of network designers, operators, vendors, researchers, and technologists that work on the Internet architecture, and its techniques and protocols. An IETF official technical specification or recommendation is called a Request for Comments (RFC).

The technical work of the IETF is done in its working groups, which are organized by topic into several areas, such as routing, transport and security. In the security area, different working groups develop security mechanisms for different protocols or applications. For example,

1. The PKIX (Public-Key Infrastructure X.509) Working Group (PKIX-WG) developed technical specifications and recommendations to support a Public Key Infrastructure, based on the X.509 protocol, which is used to build a trust and authentication services infrastructure,
2. The IPSEC (Internet Protocol Security) working group developed a protocol and other technical recommendations for secure routing between network devices, and
3. The TLS (Transport Layer Security) working group has been specifying a communication protocol and technical recommendations to provide security services for communication between a server and a client, etc.

NIST-approved cryptographic algorithms, such as block cipher modes of operation, hash functions, key establishment schemes, and digital signatures are used in various IETF protocols. For example, RFC 5288 specifies the AES Galois Counter Mode (GCM) Cipher Suites for TLS, based on SP 800-38D.

[19] IEEE P1363.2, *Password-Based Public-Key Cryptography*.
[20] IEEE 802.11, *Wireless Local Area Networks*.
[21] IEEE P1619, *Standard for Cryptographic Protection of Data on Block-Oriented Storage Devices*.

Further information is available at the IETF web site, http://ietf.org.

2.3.4 International Organization for Standardization (ISO)[22]

ISO is a non-governmental, worldwide federation of national standards bodies. Its mission is to develop international standards that help to make industry more efficient and effective. ISO standards cover almost all aspects of technology and business, from food safety to computers, and from agriculture to healthcare. Experts from all over the world develop the standards that are required by their sector, using a consensus process.

ISO/IEC JTC 1 is a joint technical committee of the International Organization for Standardization (ISO) and the International Electrotechnical Commission (IEC). ISO/IEC JTC 1 SC 27 is the subcommittee for IT security. Working group 2 (WG2) is the group developing standards for cryptography and security mechanisms. It usually has more than twenty active projects to develop either a revision of an existing standard or a new standard. Each standard consists of multiple parts, and each part includes multiple algorithms and/or mechanisms.

The cryptographic algorithms and schemes in FIPS and SPs are usually included in ISO/IEC JTC 1 standards, along with many other algorithms submitted by other countries. The following is a list of ISO/IEC standards that include cryptographic algorithms and schemes specified in NIST standards.

1. ISO/IEC 9797-1:2011, *Information technology – Security techniques – Message Authentication Codes (MACs) -- Part 1: Mechanisms using a block cipher*.
2. ISO/IEC 9797-2:2011, *Information technology – Security techniques – Message Authentication Codes (MACs) -- Part 2: Mechanisms using a dedicated hash-function*.
3. ISO/IEC 10116:2006, *Information technology – Security techniques – Modes of operation for an n-bit block cipher*.
4. ISO/IEC 10118-3:2004, *Information technology – Security techniques – Hash-functions -- Part 3: Dedicated hash-functions*.
5. ISO/IEC 11770-3:2008, *Information technology – Security techniques – Key management -- Part 3: Mechanisms using asymmetric techniques*.
6. ISO/IEC CD 11770-6, *Information technology – Security techniques – Key management -- Part 6: Key derivation*.
7. ISO/IEC 14888-2: 2008, *Information technology – Security techniques – Digital signatures with appendix -- Part 2: Integer factorization based mechanisms*.
8. ISO/IEC CD 14888-3, *Information technology – Security techniques – Digital signatures with appendix -- Part 3: Discrete logarithm based mechanisms*.
9. ISO/IEC 18033-3:2010, *Information technology – Security techniques – Encryption algorithms – Part 3: Block ciphers*.

[22] Further information is available at the ISO web site, http://www.iso.org.

10. ISO/IEC 19772:2009, *Information technology – Security techniques – Authenticated encryption*.

2.3.5 Trusted Computing Group (TCG)

The Trusted Computing Group (TCG) develops and promotes a set of industry standards that build upon roots of trust. Roots of Trust (RoTs) are hardware, firmware, and software components that are inherently trusted to perform specific, vital security functions. Because misbehavior by RoTs cannot be detected, they must be secure by design. To ensure that they are reliable and resistant to tampering, RoTs are often implemented in, or protected by, hardware.

Industry standards developed by the TCG define the capabilities of a set of fundamental roots of trust, and describe how to use those roots of trust in a variety of architectures and use cases. Many of the use cases supported by TCG technologies and specifications focus on one or more of the following areas: 1) device identity, 2) cryptographic key or credential storage, and 3) attestation of the system state.

Technologies supporting TCG-developed standards are deployed enterprise-class clients and servers, storage devices, embedded systems, and virtualized devices. Families of relevant TCG standards and specifications include:

- Trusted Platform Module (TPM): A TPM is a cryptographic module that can, among other capabilities, establish device identity in a platform, provide secure storage for keys and credentials, and support the measurement and reporting of the system state. The TPM 2.0 Library Specification provides the general architecture and command set for TPMs, with platform-specific specifications detailing how a TPM can be implemented in particular classes of systems. ISO/IEC JTC 1 has approved the TPM Library Specification as ISO/IEC 11889:2015 Parts 1-4.
- Trusted Network Connect (TNC): The TCG's TNC Working Group defines specifications that allow network administrators to enforce policies regarding endpoint integrity on devices connected to a network. These specifications were the basis for much of the work in the IETF's Network Endpoint Assessment (NEA) working group, and are highly complimentary to the on-going work in the IETF Security Automation and Continuous Monitoring (SACM) working group.
- Storage: The TCG's Storage Work Group defines specifications that enable standards-based mechanisms to protect data on storage devices, and manage these devices and capabilities. The TCG's storage specifications break out from a common core specification into two Security Subsystem Classes (SSCs): the Opal SSC, which is intended for client devices (e.g., tablets, notebooks and desktops), and the Enterprise SSC, which is intended for high-performance storage systems (e.g., servers).

SECTION 3: CRYPTOGRAPHIC ALGORITHMS

This document describes three types of cryptographic algorithms: cryptographic hash functions, symmetric-key algorithms and asymmetric-key algorithms, which are discussed in Sections 3.1, 3.2 and 3.3, respectively. Other topics to be introduced in this section include the concept of algorithm security strength and algorithm lifetime (see Sections 3.4 and 3.5, respectively).

3.1 Cryptographic Hash Functions

A hash function (also called a hash algorithm) is a cryptographic primitive algorithm that produces a condensed representation of its input (e.g., a message). A hash function takes an input of arbitrary length and outputs a value with a predetermined length. Common names for the output of a hash function include *hash value* and *message digest*.

A cryptographic hash function is a one-way function that is extremely difficult to invert. That is, it is not practical to reverse the process from the hash value back to the input.

Figure 1 depicts the process of generating and verifying a hash value.

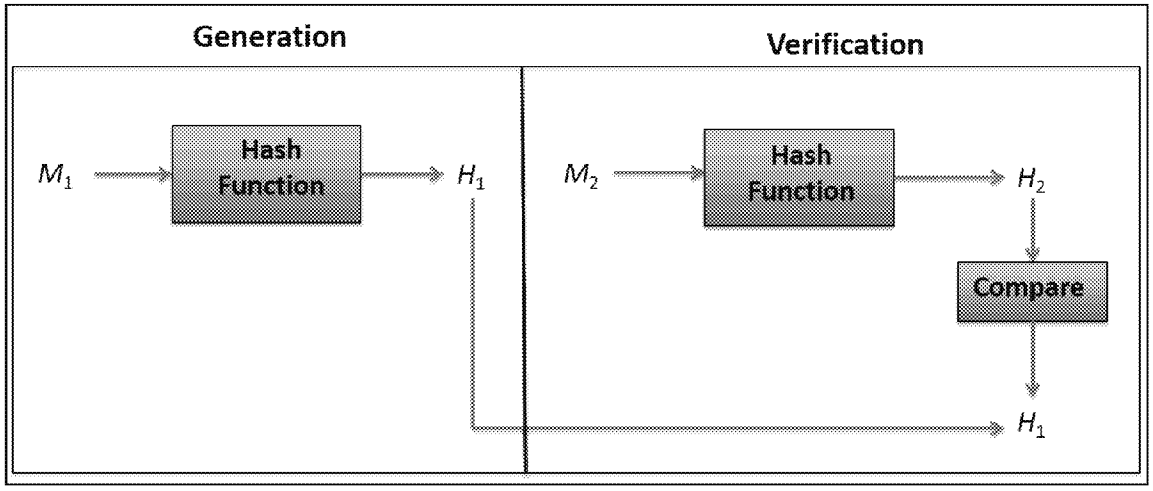

Figure 1: Hash Function Generation and Verification

A hash function is used as follows:

- Hash Generation:
  1. Hash value (H_1) is generated on data (M_1) using the hash function.
  2. M_1 and H_1 are then saved or transmitted.
- Hash Verification:
  1. Hash value (H_2) is generated on the received or retrieved data (M_2) using the same hash function that generated H_1.
  2. H_1 and H_2 are compared. If $H_1 = H_2$, then it can be assumed that M_1 has not changed during storage or transmission.

The above description is for the simplest use of a hash function. Hash functions are usually used in higher-level algorithms, including:

- Keyed-hash message authentication code algorithms (Sections 3.2.2 and 4.2.2.2),
- Digital signature algorithms (Section 4.2.3),
- Key derivation functions (e.g., for key establishment) (Section 5.3.2), and
- Random bit generators (Section 4.4).

Approved hash functions for Federal Government use are specified in FIPS 180[23] and FIPS 202[24].

- FIPS 180 specifies the SHA-1, SHA-224, SHA-256, SHA-384, SHA-512, SHA-512/224 and SHA-512/256 hash functions. Additional guidance for the use of these hash functions is provided in SP 800-106[25] and SP 800-107[26].

  Note that attacks on SHA-1 have indicated that SHA-1 provides less security than originally thought when generating digital signatures (see Section 4.2.3); consequently, SHA-1 is now disallowed for that purpose. However, SHA-1 may continue to be used for most other hash-function applications, including the verification of digital signatures previously signed using SHA-1 as the hash function (see SP 800-131A[27]).

- FIPS 202 specifies SHA3-224, SHA3-256, SHA3-384 and SHA3-512. This FIPS also specifies two extendable-output functions (SHAKE128 and SHAKE256), which are not, in themselves, considered to be hash functions; guidance on their use will be provided in the future.

3.2 Symmetric-Key Algorithms

Symmetric-key algorithms (sometimes called secret-key algorithms) use a single key to both apply cryptographic protection and to remove or check the protection. For example, the key used to encrypt data (i.e., apply protection) is also used to decrypt the encrypted data (i.e., remove the protection); in the case of encryption, the original data is called the plaintext, while the encrypted form of the data is called the ciphertext. The key must be kept secret if the data is to remain protected.

Several classes of symmetric-key algorithms have been approved: those based on block cipher algorithms (e.g., AES) and those based on the use of hash functions (e.g., a keyed-hash message authentication code based on SHA-1).

---

[23] FIPS 180, *Secure Hash Standard (SHS)*.

[24] FIPS 202, *SHA-3 Standard: Permutation-Based Hash and Extendable Output Functions*.

[25] SP 800-106, *Randomized Hashing for Digital Signatures*.

[26] SP 800-107, *Recommendations for Applications Using Approved Hash Algorithms*.

[27] SP 800-131A, *Transitions: Recommendation for Transitioning the Use of Cryptographic Algorithms and Key Lengths*.

Symmetric-key algorithms are used for:

- Encryption to provide data confidentiality (see Section 4.1),
- Authentication to provide assurance of data integrity and the source of the data (see Section 4.2),
- Key derivation (see Section 5.3.2),
- Key wrapping (see Section 5.3.5), and
- Random bit generation (see Section 4.4).

When using a symmetric-key algorithm, a unique key needs to be generated for each cryptographic relationship[28] and for each purpose (e.g., encryption, data integrity authentication and key wrapping). Technically, the same key can be used for multiple purposes when the same algorithm is used, but this is usually ill-advised, as the use of the same key for two different cryptographic processes (e.g., HMAC and key derivation using the same hash function) may weaken the security provided by one or both of the processes. However, exceptions to this rule have been approved (see Section 4.3).

As an example of the number of keys required for the use of symmetric-key algorithms, suppose that there are four entities (A, B, C, and D) that need to communicate using encryption, with each pair of entities using a different encryption key. There are six possible pair-wise relationships (A-B, A-C, A-D, B-C, B-D, and C-D), so, at least six keys are required[29]. If, instead, there are 1000 entities that wish to communicate with each other, there are 499 500 possible pair-wise relationships, and at least one unique key would be required for each relationship. If more than one algorithm, key length or purpose is to be supported (e.g., both encryption and key wrapping), then additional keys will be needed. Each entity must keep all its symmetric keys secret and protect their integrity. The need for a large number of keying relationships is a significant problem; methods for mitigating this problem are discussed in Section 5.

Several symmetric-key algorithms have been **approved** by NIST for the protection of sensitive data. However, some of these algorithms are no longer approved for applying cryptographic protection (e.g., encryption), but may continue to be used for processing already-protected information (e.g., decryption), providing that the risk of doing so is acceptable (e.g., there is reason to believe that a key was not compromised). See SP 800-57, Part 1 and SP 800-131A for more information about the acceptability of using the different cryptographic algorithms.

[28] A cryptographic relationship exists when two or more parties can communicate using the same key and algorithm. A relationship may be one-to-one or one-to-many (e.g., broadcast).

[29] Although only six cryptographic relationships are used in the example, different keys may be required by some protocols for each communication direction, i.e., a different key may be required for communications sent from A to B than is used for communications sent from B to A.

3.2.1 Block Cipher Algorithms

A block cipher algorithm is used with a single key in an **approved** mode of operation to both apply cryptographic protection (e.g., encrypt) and to subsequently process the protected information (e.g., decrypt). Several block cipher algorithms have been approved by NIST as cryptographic primitives, some of which may no longer be approved for applying cryptographic protection. However, they may still be needed for processing information that was previously protected (e.g., they may be needed for decrypting previously encrypted information).

The block cipher algorithms are discussed in Sections 3.2.1.1 through 3.2.1.4. The **approved** modes of operation are discussed in Section 3.2.1.5.

3.2.1.1 Data Encryption Standard (DES)

The Data Encryption Standard (DES) became effective in July 1977, and was the first NIST-**approved** cryptographic algorithm. It was reaffirmed several times, but due to advances in computer power and speeds, the strength of the DES algorithm is no longer sufficient to adequately protect Federal Government information. Therefore, DES was withdrawn as an **approved** algorithm in 2005 (i.e., the use of DES is no longer approved for encryption or otherwise applying cryptographic protection). However, the DES "cryptographic engine" continues to be used as a component function of TDEA (see the next section).

3.2.1.2 Triple Data Encryption Algorithm (TDEA)

The Triple Data Encryption Algorithm (TDEA), also known as Triple DES, uses the DES cryptographic engine to transform data in three operations. TDEA is specified in SP 800-67.

TDEA encrypts data in blocks of 64 bits, using three keys that define a key bundle. The use of TDEA using three distinctly different (i.e., mathematically independent) keys is **approved** and is commonly known as three-key TDEA (also referred to as 3TDEA or 3TDES).

Other variations of TDEA, where two or three of the keys are identical, are no longer approved for applying cryptographic protection because of increased computing power or weaknesses in the algorithm.

3.2.1.3 SKIPJACK

SKIPJACK is referenced in FIPS 185[30] and specified in a classified document. SKIPJACK is no longer considered adequate for the protection of federal information and has been withdrawn as a FIPS. The use of SKIPJACK for applying cryptographic protection (e.g., encryption) is **not approved**, although it is permissible to use the algorithm for decrypting information.

[30] FIPS 185, *Escrowed Encryption Standard.*

3.2.1.4 Advanced Encryption Standard (AES)

The Advanced Encryption Standard (AES) was developed as a replacement for DES and is the preferred block cipher algorithm for new products. AES is specified in FIPS 197. AES operates on 128-bit blocks of data, using 128-, 192- or 256-bit keys.

Note that the performance of AES is significantly better than that of TDEA.

3.2.1.5 Modes of Operation

With a symmetric-key block cipher algorithm, the same input block will always produce the same output block when the same key is used. Furthermore, certain kinds of data patterns in the plaintext, such as repeated blocks, would be apparent in the ciphertext. To counteract these properties, modes of operation have been specified to use a block cipher algorithm to provide an information service, such as confidentiality or integrity protection.

These modes combine the cryptographic primitive algorithm with a symmetric key and variable starting values (commonly known as initialization vectors) to perform a cryptographic service (e.g., the encryption of a message). **Approved** modes for block cipher algorithms have been specified in the SP 800-38 series of publications and include modes for:

- Encryption, as specified in SP 800-38A, SP 800-38E and SP 800-38G (see Section 4.1),
- Authentication, as specified in SP 800-38B (see Section 4.2.2.1),
- Authenticated encryption, as specified in SP 800-38C and SP 800-38D (see Section 4.3), and
- Key wrapping, as specified in SP 800-38F (see Section 5.3.5).

3.2.2 Hash-based Symmetric-key Algorithms

A symmetric-key algorithm based on the use of a hash function has been specified in FIPS 198[31]. This algorithm, known as HMAC, has been **approved** for use with any **approved** hash function specified in FIPS 180 or FIPS 202. Guidance on the use of the hash functions specified in FIPS 180 for HMAC is provided in SP 800-107.

3.3 Asymmetric-Key Algorithms

Asymmetric-key algorithms (often called public-key algorithms) use a pair of keys (i.e., a key pair): a public key and a private key that are mathematically related to each other. The public key may be made public without reducing the security of the process, but the private key must remain secret if the data is to retain its cryptographic protection. Even though there is a relationship between the two keys, the private key cannot efficiently be determined based on knowledge of the public key.

[31] FIPS 198, *Keyed Hash Message Authentication Code (HMAC)*.

One of the keys of the key pair is used to apply cryptographic protection, and the other key is used to remove or verify that protection. The key to use depends on the algorithm used and the service to be provided. For example, a digital signature is computed using a private key, and the signature is verified using the public key (i.e., the protection is applied using the private key and verified using the corresponding public key). For those asymmetric algorithms also capable of encryption[32], the encryption is performed using the public key, and the decryption is performed using the private key (i.e., the protection is applied using the public key and removed using the private key).

Asymmetric-key algorithms are used primarily for data integrity authentication and source authentication (see Section 4.2), and for key establishment (see Section 5.3). These algorithms tend to be much slower than symmetric-key algorithms, so are not used to process large amounts of data. However, when used for key establishment (see Section 5), there are methods that combine the use of symmetric and asymmetric algorithms to reduce the number of keys required for establishing cryptographic relationships.

Key pairs for asymmetric-key algorithms should be generated for each purpose (e.g., one key pair for generating and verifying digital signatures, and a different key pair for key establishment). Technically, it is sometimes possible to use the same key pair for more than one purpose, but this is ill-advised, as the use of the same key pair for two different cryptographic purposes (e.g., digital signatures and key establishment) may weaken the security provided by one or both of the processes.

The use of asymmetric-key algorithms requires the establishment of fewer initial keys than the use of symmetric-key algorithms. As an example, suppose that an entity wants to generate digital signatures and participate in a key-establishment process using its own key pair[33]; a key pair needs to be generated for each purpose. If there are six entities that intend to both generate digital signatures and participate in the key-establishment process, then six key pairs are needed for digital signature generation, and another six key pairs are needed for key establishment, for a total of twelve key pairs. For 1000 entities, 1000 key pairs of each would be needed for each purpose, for a total of 2000 key pairs. A unique key pair does not need to be generated for each relationship; recall that for symmetric-key algorithms, a unique key needs to be generated for each relationship (see Section 3.2). If multiple public-key algorithms or key lengths are to be used for either process, then additional key pairs will be required.

The private key is retained by the entity who "owns" the key pair; it must be kept secret and its integrity protected. The public key is usually distributed to other entities and requires integrity protection; this is often accomplished by using a public-key certificate, as discussed in Section 5.2.3. When a public-key certificate is used, the certificate provides the integrity protection for the public key, so the burden of key protection by each entity is limited to only those private keys owned by the entity.

---

[32] Not all public-key algorithms are capable of multiple functions, e.g., both encryption and decryption, and the generation and verification of digital signatures.

[33] Note that some key-establishment schemes do not require that all parties have key pairs, so some parties will not need a key pair for key establishment.

Some asymmetric-key algorithms use domain parameters, which are additional values necessary for the use of the cryptographic algorithm. These values are mathematically related to each other and to the keys with which they will be used. Domain parameters are usually public and are used by a community of users for a substantial period of time. These domain parameters are either contained within or referenced by a certificate containing a public key.

The secure use of asymmetric-key algorithms is dependent on users obtaining certain assurances:

- Assurance of domain-parameter validity (for those algorithms requiring domain parameters) provides confidence that the domain parameters are mathematically correct,
- Assurance of public-key validity provides confidence that the public key appears to be a suitable key, and
- Assurance of private-key possession provides confidence that the party that is supposedly the owner of the private key really has the key.

3.3.1 DSA

The Digital Signature Algorithm (DSA) is **approved** and specified in FIPS 186. This algorithm is used to generate and verify digital signatures using finite-fields. FIPS 186 defines methods for generating DSA domain parameters and key pairs, and specifies the key lengths to be used for secure interoperability and the algorithms to be used for digital-signature generation and verification.

3.3.2 ECDSA

The Elliptic Curve Digital Signature Algorithm (ECDSA) is **approved** within FIPS 186, but actually specified within American National Standard (ANS) X9.62[34]. The basic signature and verification algorithms are the same as those used for DSA, except that the mathematics is based on the use of elliptic curves, rather than finite fields. FIPS 186 provides guidance for the use of ECDSA within the Federal Government, as well as providing recommended elliptic curves to facilitate interoperability and security. An advantage of using ECDSA is that the key lengths are considerably shorter than those used for DSA and RSA, requiring less storage space and transmission bandwidth, and the execution of the algorithm is generally faster than DSA and RSA

ANS X9.62 includes specifications for the generation of the ECDSA domain parameters and key pairs, as well as the algorithms for digital signature generation and verification. FIPS 186 defines the key lengths to be used for secure interoperability, provides additional guidance on the use of random bit generators to generate the key pairs, and recommends elliptic curves for use by the Federal Government. Note that the same elliptic curves are also included in ANS X9.62.

---

[34] ANS X9.62, *Public Key Cryptography for the Financial Services Industry: The Elliptic Curve Digital Signature Algorithm (ECDSA)*.

3.3.3 RSA

The RSA algorithm is **approved** for the generation and verification of digital signatures in FIPS 186] and specified in PKCS 1[35] and ANS X9.31[36]. FIPS 186 includes restrictions on the use of RSA to generate digital signatures, methods to generate RSA key pairs, and defines the key lengths to be used for secure interoperability.

The RSA primitive can be used for key establishment, as well as for the generation and verification of digital signatures. Its use for key establishment is specified in SP 800-56B[37]; that publication specifies **approved** methods for both key agreement and key transport (see Section 5.3 for further information on key establishment, key agreement and key transport).

The key pairs used for RSA digital-signature generation and verification, and for RSA key establishment are generated in the same way, but need to be different for each purpose.

3.3.4 Diffie-Hellman and MQV

Diffie-Hellman (DH) and MQV[38] are two classes of key-establishment algorithms used for key agreement (see Section 5.3.3). The use of these algorithms for key agreement is specified in SP 800-56A[39] using both finite-fields and elliptic-curves. For elliptic-curve key pairs and domain parameters, the methods for generating those key pairs and domain parameters are specified in ANS X9.62 using the same methods used to generate ECDSA key pairs and domain parameters.

The recommended elliptic curves for elliptic-curve DH and MQV are the same as those provided in FIPS 186 for ECDSA.

3.4 Algorithm Security Strength

The security strength of a cryptographic algorithm is measured by an attacker's difficulty in breaking the algorithm. Breaking a cryptographic algorithm can be defined as defeating some aspect of the protection that the algorithm is intended to provide. For example, a block cipher encryption algorithm that is used to protect the confidentiality of data is broken if, with an acceptable amount of work, it is possible to determine the value of its key or to recover the plaintext from the ciphertext without knowledge of the key.

---

[35] Public Key Cryptography Standard #1.

[36] ANS X9.31, *Digital Signatures Using Reversible Public Key Cryptography For The Financial Services Industry (RDSA)*. This standard has been withdrawn as an ANSI standard.

[37] SP 800-56B, *Recommendation for Pair-wise Key Establishment Schemes Using Integer Factorization Cryptography*.

[38] Menezes–Qu–Vanstone.

[39] SP 800-56A, *Recommendation for Pair-Wise Key-Establishment Schemes Using Discrete Logarithm Cryptography*.

SP 800-57, Part 1 provides the current estimates for the security strengths that can be provided by the **approved** cryptographic algorithms; these strengths have been determined with respect to specific key lengths.

The **approved** security strengths for federal applications are 112, 128, 192 and 256 bits. Note that a security strength of 80 bits was previously approved as well. Since it is no longer considered as providing adequate protection, the use of algorithms and keys providing a security strength of 80 bits is **no longer approved** for applying cryptographic protection (e.g., encrypting data). However, algorithms and keys providing 80 bits of strength can be used for processing data that was previously protected at that strength (e.g., for decryption).

Appropriate algorithms, key lengths, and key generation and handling methods need to be used to actually support those security strengths, and are further discussed in Section 5.1.4.

3.5 Algorithm Lifetime

Over time, algorithms may be successfully attacked so that the algorithm no longer provides the desired protection. The attack could be on the algorithm itself, or could be on the algorithm with a specific key length. In the latter case, the use of a longer key may prevent a successful attack, or at least delay it for a period of time.

When selecting the algorithms and key lengths to be used for an application, the length of time for which the data needs to be protected should be taken into account so that a suitable algorithm and key length is used. SP 800-57, Part 1 provides a current estimate of the time frames during which the **approved** algorithms and key lengths are considered to be secure. The algorithms and key lengths used for cryptographic protection need to fall within the estimated time frame. However, these estimates are just that – estimates. It is possible that an advance in technology (e.g., the use of quantum computers and algorithms) or cryptanalysis could occur prior to the end date of that time frame. It is often the case that these advances are initially impractical or limited in their threat. It is recommended that an organization have a transition strategy for addressing this problem if it occurs, including assessing the risk for the compromise of the organization's data, and transitioning to a new algorithm or key length, as appropriate.

SECTION 4: CRYPTOGRAPHIC SERVICES

All sensitive information requires integrity protection, and confidentiality protection may be required as well. This section discusses the cryptographic services that can be provided for the protection of sensitive data other than keys. These services include data confidentiality, data integrity authentication and source authentication, including non-repudiation. The protection and management of the keys used while providing these cryptographic services are discussed in Section 5.

Ideally, cryptographic services would be provided using as few algorithms as possible. For example, AES could be used to provide confidentiality (Section 4.1), data integrity authentication (Section 4.2), key wrapping (Section 5.3.5) and as the basis for a random bit generator (see Section 4.4). However, this may not be as practical as it first appears, as other algorithms may also be available that are needed for different applications and that provide other security properties.

4.1 Data Confidentiality

Encryption is used to provide confidentiality for data. The unprotected form of the data is called plaintext. Encryption transforms the data into ciphertext, and ciphertext can be transformed back into plaintext using decryption. Data encryption and decryption are generally provided using symmetric-key block cipher algorithms. The **approved** symmetric-key algorithms for data encryption are: AES and TDEA (see Section 3.2.1.4 and Section 3.2.1.2, respectively). Decryption of the ciphertext is performed using the algorithm and key that were used to encrypt the plaintext. Unauthorized recipients of the ciphertext who know the cryptographic algorithm but do not have the correct key should not be able to decrypt the ciphertext. However, anyone who has the key and the cryptographic algorithm can easily decrypt the ciphertext and obtain the original plaintext.

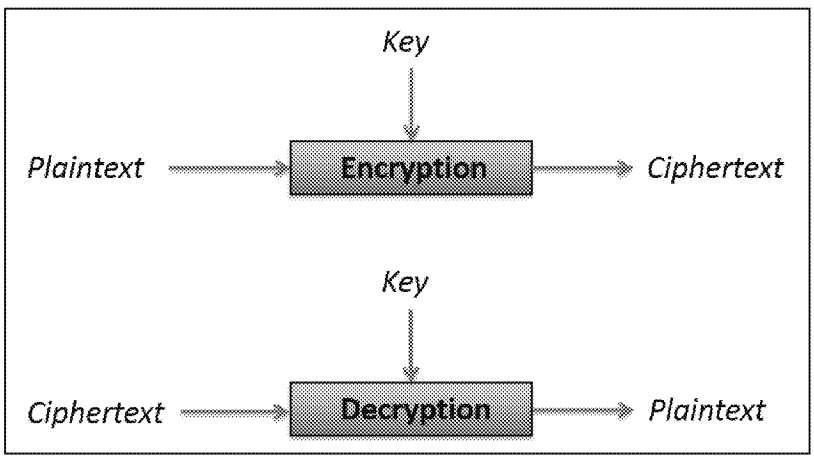

Figure 2: Encryption and Decryption

Figure 2 depicts the encryption and decryption processes. The plaintext and a key are used by the encryption process to produce the ciphertext. To decrypt, the ciphertext and the same key are used by the decryption process to recover the plaintext data.

Note that asymmetric-key algorithms could also be used to encrypt and decrypt data, but because these algorithms are slow in comparison to block cipher algorithms, they are not normally used to encrypt and decrypt general data; they can, however, be used to protect keys, as discussed in Section 5.

As discussed in Section 3.2.1.5, encryption is performed using a block cipher algorithm and a mode of operation. The **approved** modes of operation for encryption are specified in:

- SP 800-38A for AES and TDEA: the Electronic Codebook (ECB), Cipher Block Chaining (CBC), Cipher Feedback (CFB), Counter (CTR), and Output Feedback (OFB) modes,
- SP 800-38E for AES: the XTS-AES mode (for protecting the confidentiality of data on storage devices only), and
- SP 800-38G for AES: the FF1 and FF3 modes for Format Preserving Encryption.

Additional modes that provide both confidentiality and authentication (as discussed in Section 4.2) are discussed in Section 4.3.

4.2 Data Integrity and Source Authentication

Data integrity (often referred to as simply *integrity*) is concerned with whether or not data has changed between two specified times (e.g., between the time when the data was created, stored and/or transmitted, and the time when it was retrieved and/or received). While data integrity cannot be guaranteed, the use of data integrity codes provides a means to detect changes with a high probability. A data integrity code is computed on data when it is created, before storage or before transmission, and computed again when the data is retrieved or received. Verification that these computations agree provides a measure of assurance of data integrity. In cryptographic literature, this process is called *message* (or data) *authentication*.

Source authentication is a process used to provide assurance of the source of information. Source authentication includes identity authentication, which provides assurance to one of the parties in a communication (say, Bob) that he is receiving data from or providing data to another specific party (say, Alice). Depending on the method used, source authentication could also support non-repudiation, whereby both Bob and some third party (say, Carl) have some assurance that the data came from Alice.

Cryptography can be used to provide these services, but the same algorithm may not provide all of them. Hash functions, as discussed in Section 4.2.1, can be used to provide some assurance of data integrity. Message Authentication Code (MAC) algorithms, as discussed in Section 4.2.2, can provide both data integrity and source authentication services. Digital signature algorithms can be used to provide data integrity and source authentication services, as well as supporting non-repudiation, but at a higher performance cost (see Section 4.2.3).

4.2.1 Hash Functions

A hash function is used to generate a hash value that can provide some assurance of the integrity of the data over which the hash value is generated. However, if a hash function is used alone (e.g., without the use of a secret key, as is the case of HMAC, or in conjunction with the generation of digital signatures), there is no assurance that the data has not been altered by an adversary and a new hash value computed. Therefore, the use of a hash function alone for providing integrity protection is not recommended unless there is a very low risk of this scenario (e.g., when data is provided by a trusted source, and the hash value is used only to determine changes that may occur because of a degraded transmission medium).

4.2.2 Message Authentication Code Algorithms

A Message Authentication Code algorithm and a cryptographic key are used to generate a message authentication code (MAC) that can be used to provide assurance of data integrity and source authentication. A MAC is a cryptographic checksum on the data that can provide assurance that the data has not changed or been altered since some point in time, and that the MAC was computed by the party or parties sharing the key. Typically, MACs are used between two or more parties that share the same secret key to authenticate information exchanged between those parties; the use of MACs to provide data integrity and source authentication depends on limiting knowledge of the secret key to only those parties. Since a MAC key is shared among a community of users (e.g., two or more parties), only those parties sharing the key can compute a correct MAC on given data.

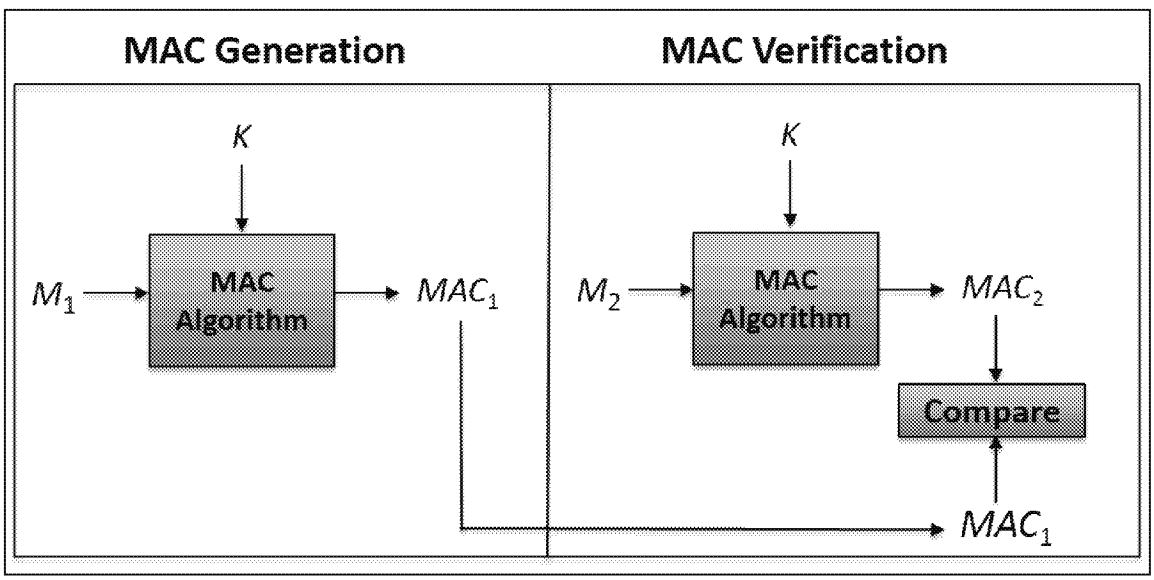

Figure 3: Message Authentication and Verification

Figure 3 depicts the use of MACs:

- A MAC (MAC_1) is computed on data (M_1) using a key (K). M_1 and MAC_1 are then saved or transmitted.

- At a later time, the integrity of the retrieved or received data is checked by labeling the retrieved or received data as M_2 and computing a MAC (MAC_2) on M_2 using the same key (K).
- If MAC_1 is the same as MAC_2, then it can be assumed that M_2 is the same as the original data (M_1) (i.e., $M_1 = M_2$). The verifying party also knows that only a party that shares the key could have correctly generated the MAC.

For example, if two parties (e.g., A and B) share a key, party A generates the MAC and sends it to party B, and party B successfully verifies the received MAC, then party B knows that party A generated the original MAC, and source authentication has been accomplished. However, if three parties share the key (e.g., A, B and C), party A generates the MAC to be sent to party B, and party B successfully verifies the received MAC; party B knows that either party A or party C generated the original MAC, but has no proof of which one. Note that this may be acceptable for some applications.

MACs are used to detect data modifications that occur between the initial generation of the MAC and the verification of the received MAC. They do not detect errors that occur before the MAC is originally generated.

Assurance of data integrity is frequently provided using non-cryptographic techniques known as error detection codes. However, these codes can be altered by an adversary to the adversary's benefit. The use of an **approved** cryptographic mechanism, such as a MAC, addresses this problem. That is, the assurance of integrity provided by a MAC is based on the assumption that it is not likely that anyone could correctly generate a MAC without knowing the cryptographic key. An adversary without knowledge of the key will be unable to modify data and then generate a verifiable MAC on the modified data. It is therefore crucial that MAC keys be kept secret.

Two types of algorithms for computing a MAC have been **approved** for Federal Government use: MAC algorithms that are based on symmetric-key block cipher algorithms, and MAC algorithms that are based on hash functions.

4.2.2.1 MACs Based on Block Cipher Algorithms

The SP 800-38 series of publications includes modes for the generation of MACs:

- SP 800-38B[40] defines the CMAC mode for computing a MAC using the NIST-**approved** block-cipher algorithms: AES and TDEA.
- SP 800-38D[41] defines the GMAC mode for the computation of a MAC using AES.
- Modes providing both confidentiality (i.e., encryption) and authentication (i.e., computing a MAC) in a single operation are also defined (see Section 4.3).

[40] SP 800-38B, *Recommendation for Block Cipher Modes of Operation: The CMAC Mode for Authentication*.

[41] SP 800-38D, *Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC*.

4.2.2.2 MACs Based on Hash Functions

FIPS 198[42] defines a MAC (HMAC) that uses a cryptographic hash function in combination with a secret key. HMAC must be used with an **approved** cryptographic hash function (see Section 4.2.1). The security associated with the use of HMAC is discussed in SP 800-107[43].

4.2.3 Digital Signature Algorithms

A digital signature algorithm is used with a pair of keys – a private key and a public key – to generate and verify digital signatures. The private key is used to generate signatures and must be known only by the signer (the key-pair owner); the public key is used to verify the signatures. Because of the design of the algorithm, and the methods for generating key pairs, the public key cannot efficiently be used to determine the private key. Because two keys are required for the generation and verification process, digital signature algorithms are classified as asymmetric-key algorithms.

A digital signature is represented in a computer as a string of bits and is an electronic analogue of a hand-written signature that can be verified by anyone with access to the public key. The signature can be used to provide assurance of data integrity and source authentication, and to support non-repudiation.

Each signer possesses a private and public key pair. Signature generation (with a verifiable digital signature) can be performed only by the party that has access to the private key. Anyone that knows the public key can verify the signature by employing the associated public key. The security of a digital-signature system is dependent on maintaining the secrecy of the signer's private key. Therefore, signers must guard against the unauthorized acquisition of their private keys.

Digital signatures offer protection that is not available by using alternative signature techniques. One such alternative is a digitized signature. A digitized signature is generated by converting a visual form of a handwritten signature to an electronic image (e.g., by scanning it into a computer). Although a digitized signature resembles its handwritten counterpart when printed, it does not provide the same protection as a digital signature. Digitized signatures can be forged and can be duplicated and appended to other electronic data; digitized signatures cannot be used to determine if information has been altered after it is signed. Digital signatures, however, are computed on each message using a private key known only by the signer. Each different message signed by the signer will have a different digital signature. Even small changes to the message will result in a different signature. If an adversary does not know the private key, the adversary cannot generate a valid signature (i.e., a signature that can be verified using the public key that corresponds to the private key used to generate the signature).

Figure 4 depicts the generation and verification of digital signatures. A digital signature algorithm includes a signature generation process and a signature verification process:

[42] FIPS 198-1, *The Keyed-Hash Message Authentication Code (HMAC)*.

[43] SP 800-107, *Recommendation for Applications Using Approved Hash Algorithms*.

- Signature generation:
  - A hash function (see Section 3.1) is used in the signature generation process to obtain a hash value, which is a condensed version of the data to be signed (i.e., shown as M_1 for signature generation in Figure 4).
  - The hash value is then input to the signature generation process, along with a private key, to generate the digital signature (shown as DS_1 in Figure 4).
  - The digital signature (DS_1) is provided to the verifier, along with the signed data (M_1).
- Signature verification: The receiver of the data and signature verifies the signature as follows:
  - The received data (M_2) is hashed using the same hash function to produce another hash value.
  - The newly computed hash value and the received signature (DS_2) are input to the signature verification process, along with the the signer's public key. The output of this process is an indication of whether or not the signature is valid or invalid for the received message (M_2).

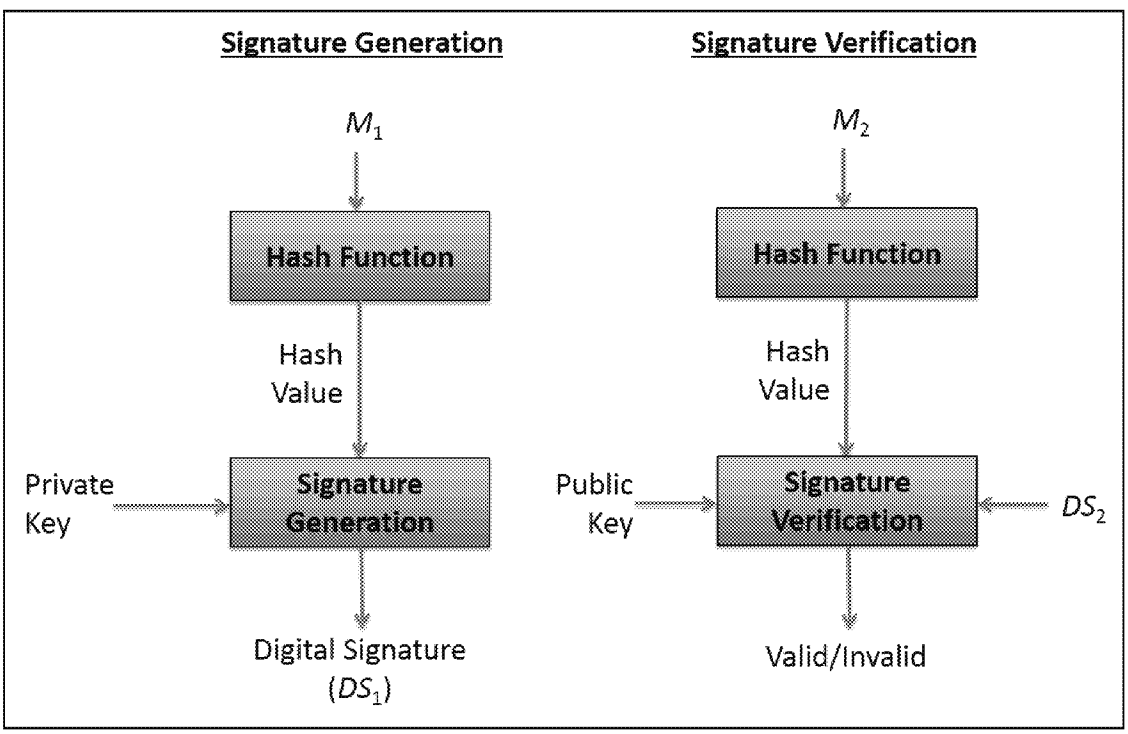

Figure 4: Digital Signature Generation and Verification

Note that the details of the signature generation and verification processes are different for each approved algorithm. Also, note that M_2 is used in the verification process rather than M_1, and DS_2 is used rather than DS_1 because of the possibility that M_1 and DS_1 could have been deliberately or accidentally modified before the verification process performed by the receiver.

FIPS 186 specifies methods for generating and verifying digital signatures using asymmetric (public-key) cryptography. The FIPS includes three digital signature algorithms:

- The Digital Signature Algorithm (DSA) (see Section 3.3.1),
- The Elliptic Curve Digital Signature Algorithm (ECDSA) (see Section 3.3.2), and
- RSA (see Section 3.3.3).

The digital signature algorithms are used in conjunction with the hash functions specified in FIPS 180[44] and FIPS 202. Each of these algorithms requires obtaining assurances about the domain parameters and/or keys used, as discussed in Section 3.3; SP 800-89[45] provides methods for obtaining these required assurances when using digital signatures.

In many cases, determining when a digital signature was generated is important. For example, it may be important to determine whether a document was signed before a certain date, e.g., which of two wills was signed closest to and prior to the date that a person died. SP 800-102[46] provides guidance on establishing when a digital signature was generated.

4.3 Combining Confidentiality and Authentication in a Block-Cipher Mode of Operation

Confidentiality and authentication can be provided using either two separate block-cipher algorithms (e.g., AES in the CBC mode for encryption and HMAC for authentication) or in a single block-cipher mode of operation. Note that in this discussion, authentication is used to obtain both an assurance of data integrity and of the source of the data that has been cryptographically protected.

If encryption and authentication are performed as two separate operations (see Sections 4.1 and 4.2, respectively), two distinct keys are required. If care is not taken in performing these operations (e.g., performing the operations in the right order), vulnerabilities can be introduced that may allow attacks.

An alternative is to use modes that both encrypt and authenticate in a single operation using a single key; such a mode is called an "authenticated-encryption" mode. Using such modes requires fewer keys and is generally faster than using two separate operations. Two authenticated-encryption modes have been defined for AES (no such mode has been defined for TDEA):

- SP 800-38C[47] specifies the CCM mode, and
- SP 800-38D[48] defines the Galois/Counter mode (GCM).

[44] FIPS 180, *Secure Hash Standard (SHS)*.

[45] SP 800-89, *Recommendation for Obtaining Assurances for Digital Signature Applications*.

[46] SP 800-102, *Recommendation for Digital Signature Timeliness*.

[47] SP 800-38C, *Recommendation for Block Cipher Modes of Operation: the CCM Mode for Authentication and Confidentiality*.

4.4 Random Bit Generation

Cryptography and security applications make extensive use of random numbers and random bits. For cryptography, random values are needed to generate cryptographic keys. The term "entropy" is used to describe the amount of randomness in a value, and the amount of entropy determines how hard it is to guess that value.

There are two classes of random bit generators (RBGs): Non-Deterministic Random Bit Generators (NRBGs), sometimes called true random number (or bit) generators, and Deterministic Random Bit Generators (DRBGs), sometimes called pseudorandom bit (or number) generators. Each RBG is dependent on the use of an entropy source to provide unpredictable bits that are outside of human control; these bits are acquired from some physical source, such as thermal noise, ring oscillators or hard-drive seek times. An NRBG is dependent on the availability of new, unused entropy bits produced by the entropy source for every NRBG output. A DRBG is initially "seeded" with entropy produced by an entropy source or using an **approved** method that depends on an entropy source (e.g., an NRBG); depending on the application, the DRBG may or may not receive additional entropy during operation (e.g., by being reseeded).

Several publications have been developed or are currently under development for random-bit generation:

- SP 800-90A[49] specifies **approved** DRBG algorithms, based on the use of hash functions and block-cipher algorithms; DRBGs must be initialized from a randomness source (e.g., an entropy source or an NRBG) that provides sufficient entropy for the security strength(s) to be supported by the DRBG.
- SP 800-90B[50], which is currently under development, discusses entropy sources, including the health tests needed to determine that the entropy source has not failed and tests to estimate how much entropy that the entropy source can reliably provide.
- SP 800-90C[51] provides constructions for the design and implementation of NRBGs and DRBGs from the algorithms in SP 800-90A and the entropy sources designed in accordance with SP 800-90B. Note that the NRBGs are constructed to include a DRBG algorithm from SP 800-90A to provide a fallback capability if an entropy source failure is not immediately detected.
- SP 800-22[52] discusses some aspects of selecting and testing random and pseudorandom number generators. This document includes some criteria for

[48] SP 800-38D, *Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC*.

[49] SP 800-90A, *Random Number Generation Using Deterministic Random Bit Generator Mechanisms*.

[50] SP 800-90B, *Recommendation for the Entropy Sources Used for Random Bit Generation*.

[51] SP 800-90C, *Recommendation for Random Bit Generator (RBG) Constructions*.

[52] SP 800-22, *A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications*.

characterizing and selecting appropriate generators, discusses statistical testing and its relation to cryptanalysis and provides some recommended statistical tests. These tests may be useful as a first step in determining whether or not a generator is suitable for a particular cryptographic application. However, for federal applications, the RBGs must be validated for compliance to FIPS 140 and the appropriate parts of SP 800-90.

4.5 Symmetric vs. Asymmetric Cryptography

As discussed in Sections 3.2 and 3.3, when large numbers of cryptographic relationships are required, the number of initial symmetric keys that will be required may be significantly larger than the number of public/private key pairs required.

However, the primary advantage of symmetric-key cryptography is speed. Symmetric-key algorithms are generally significantly faster than asymmetric-key algorithms, and the keys are shorter in length for the same security strength; the key length may be an important consideration if memory for storing the keys, or the bandwidth for transporting the keys is limited. In addition, advances in cryptanalysis and computational efficiency have tended to reduce the level of protection provided by public-key cryptography more rapidly than that provided by symmetric-key cryptography. Also, in a potential post-quantum world, the currently approved asymmetric-key algorithms will not provide adequate protection.

Since asymmetric-key (i.e., public-key) cryptography requires fewer keys overall, and symmetric-key cryptography is significantly faster, a hybrid approach is often used, whereby asymmetric-key algorithms are used for the generation and verification of digital signatures and for key establishment, while symmetric-key algorithms are used for all other purposes (e.g., encryption), especially those involving the protection of large amounts of data. For example, an asymmetric-key system can be used to establish a symmetric key via a key-agreement or key-transport process (see Sections 5.3.3 and 5.3.4, respectively), after which the symmetric key is used to encrypt files or messages.

In some situations, asymmetric-key cryptography is not necessary, and symmetric-key cryptography alone is sufficient. This includes environments where secure symmetric-key establishment can take place using symmetric keys already shared between entities, environments where a single authority knows and manages all the keys, and in single-user environments.

In general, asymmetric cryptography is best suited for an open, multi-user environment.